

SwipeBite - Bi-Weekly progress

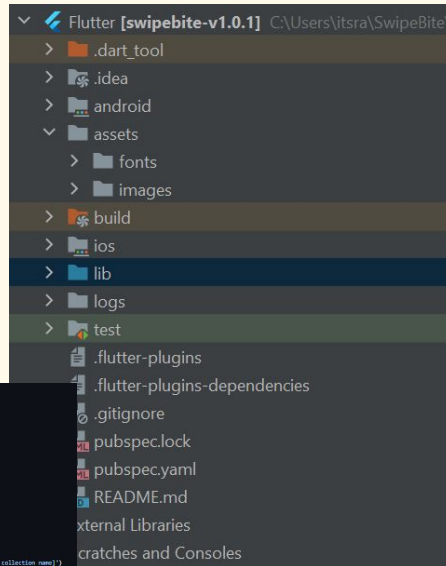
SungJun Park
Ramon Arias
Shakhzod
Yujin Juhn

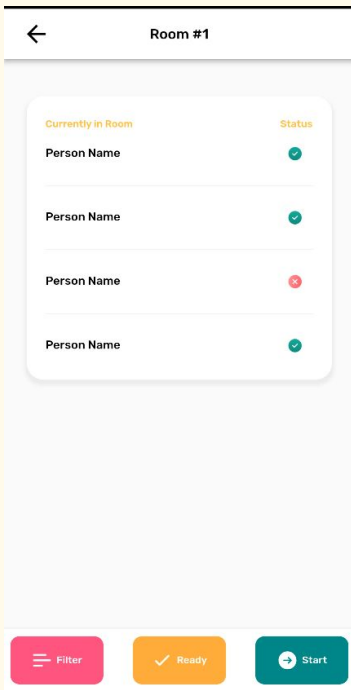


Addressing Feedback

- OS availability
- Crawling frequency
 - Exception logic
- Number of collected restaurants
 - 490 total

```
1 import sys
2 import json
3 import firebase_admin
4 from firebase_admin import credentials
5 from firebase_admin import firestore
6 import timeit
7
8 # Use a service account
9 cred = credentials.Credentials.from_service_account_info('firebase_admin_init.json')
10 firebase_admin.initialize_app(cred)
11
12 db = firestore.client()
13
14 class UploadFirestoreData:
15     def __init__(self):
16         self.start = timeit.default_timer()
17         if len(sys.argv) != 2:
18             print('Error: Check your command line arguments, 2 arguments expected (file:filepath, collectionname:firebase collection name)')
19             return None
20
21         self.json_data = sys.argv[1][1:]
22         self.collectionname = sys.argv[2][1:]
23
24     @property
25     def json_data(self):
26         return self._json_data
27
28     @json_data.setter
29     def json_data(self, val):
30         try:
31             with open(val, 'r') as f:
32                 self._json_data = json.load(f)
33         except Exception as e:
34             print(f'FILE EXCEPTION: {str(e)}')
35
36     def convert_nested_array(self, data):
37         if 'dishes' in data:
38             data['dishes'] = [{"name": dish['name'], "price": dish['price']} for dish in data['dishes']]
39         if 'reviews' in data and 'reviews' in data['reviews']:
40             data['reviews'] = [{"content": rev['content'], "count": rev['count']} for rev in data['reviews']]
41         return data
42
43     def upload(self):
44         if self.json_data:
45             for idx, item in enumerate(self.json_data):
46                 try:
47                     item = self.convert_nested_array(item)
48                     doc_id = str(timeit.default_timer()) # Assures that 'name' key exists in your data
49                     # collectionname(self.collectionname).document(doc_id).set(item)
50                     if doc_id == len(self.json_data):
51                         stop = timeit.default_timer()
52                         print('*****{}*****'.format(stop - self.start))
53                         print('Time taken "str(stop - self.start)')
54                     except Exception as e:
55                         print(f'UPLOAD EXCEPTION: {str(e)}')
56
57 uploadjson = UploadFirestoreData()
58 uploadjson.upload()
```





Front-end implementation

- Room features
- Filter implementation
- Simplified categories - 10 from 91



Front-end implementation



Front-end implementation



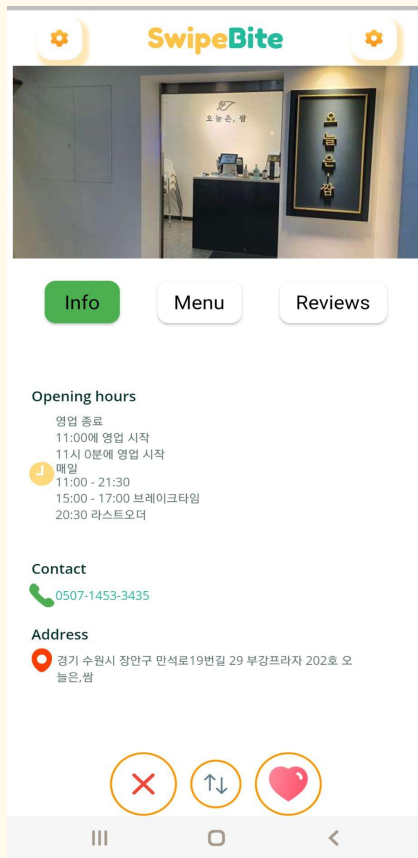
- Swipe Page has been implemented (swipe works)
 - Photos: representative photo
 - Menu plate pictures
 - 3 dish photos
 - Category
 - Number of Reviews & Parking Availability
 - DisLike / More Info / Like buttons

Front-end implementation

Post processing is needed

- workingHours : '\n' -> '\\\n'
- Reviews: '\"' needs to get removed

Front-end implementation



- More Info Page has been implemented
(Accessible by swipe or arrow button)

- Information tab: Opening Hours

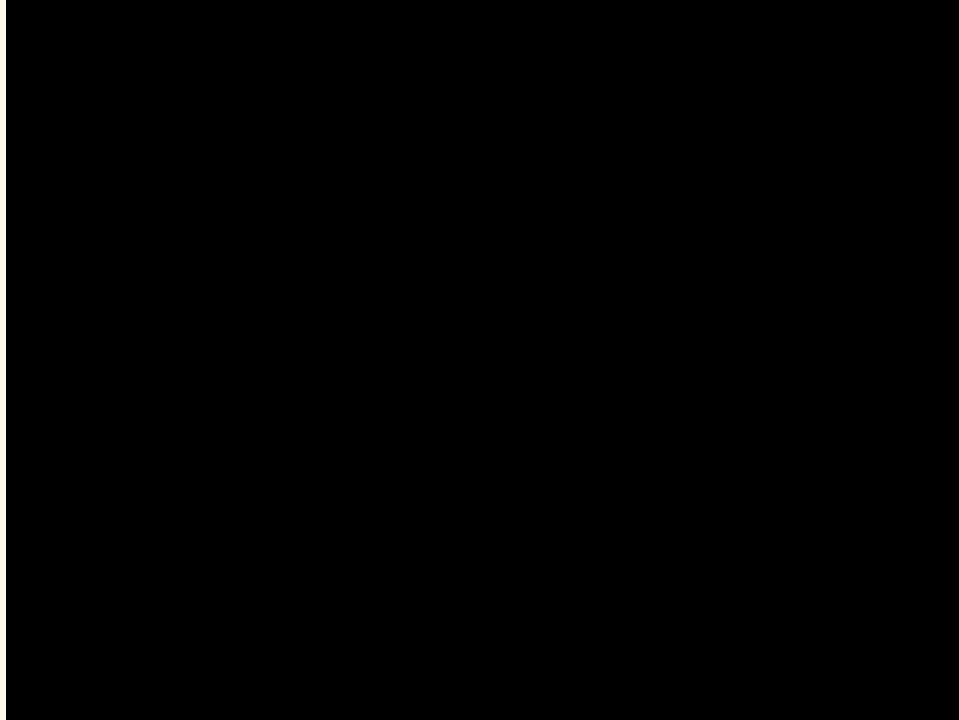
Contact

Address

- Menu tab
- Review tab:

Number of Reviews & Category of Review







Last Week



Room logic implementation

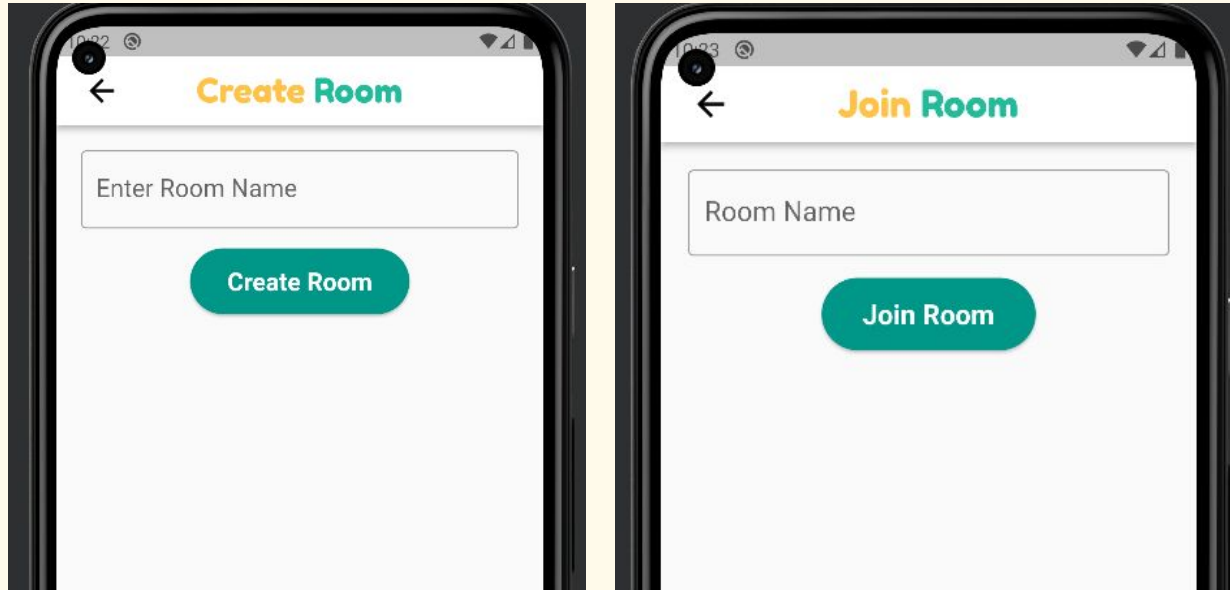
Have created new Collection in FireStore:

- Now every user may have the joined room

 (default)	 Rooms  	 4545 
+ Start collection	+ Add document	+ Start collection
RestaurantsNSC	3232	+ Add field
Rooms >	4545 >	Owner: "sdf@gmail.com"
Users	HelloRoom	<div>▼ Users</div> <div><div>0</div> "sdf@gmail.com"</div> <div><div>1</div> "shakzod42@gmail.com"</div>

Room logic implementation

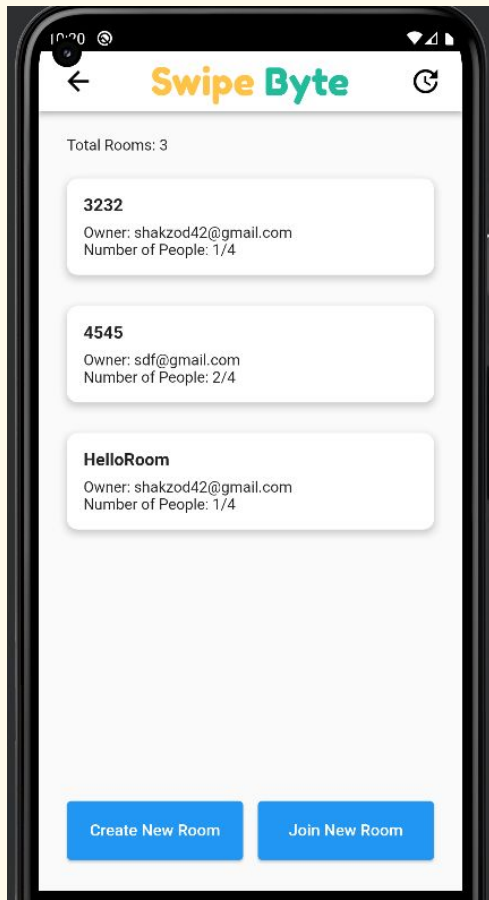
The users may join or create rooms using corresponding 2 buttons on main screen, which will show the following



Room logic implementation

Moreover, have added

- the Room information view
 - showing which rooms the user have already joined
- Update button
 - To update the room information view, when user has joined/created new room



To be done in the Following Weeks

- Attain Restaurant Information From Firebase (via Query)
- Swiping Animation needs to be implemented
- The display of the Photos needs to be fixed (some are cropped some are too zoomed in)
- Refine json data so that it would be appropriate to display ex) working hours
- Implement exception logic

Role Distribution

- **Shakhzod** - Frontend & Backend Team
 - Room logic
 - Frontend Functionality
 - Integrate Frontend
- **Sung Jun Park** - Frontend & Crawler Team
 - Naver Maps crawler- postprocess
 - Frontend Functionality
 - Integrate Frontend
- **Ramon Arias** - Frontend Team
 - Figma design
 - Frontend Functionality
 - Integrate Frontend